Optical Matched Filter Using a 4f-System

Ethan Ross

OPTI 8206 Final Project

ABSTRACT:

A matched filter 4f system was modeled and simulated in VirtualLab and MatLAB. Alphanumeric targets consisting of single characters or words were tested. The matched filter successfully located the targets in simple binary inputs. A clear peak in intensity in the output was observed at the location of the targets. The system was further tested with more realistic, noisy inputs. A photograph of the page of a book was taken and used as the input. The matched filter successfully created a peak at the target location, but additional peaks were observed at other character locations. This was likely due to complications in target definition. Finally, the matched filter was tested for a non-alphanumeric target. The system was used to attempt to locate face in Waldo in the popular *Where's Waldo* series. A small peak in intensity at the output was observed at Waldo's location. However, the peak was not as distinct as previous tests. Again this is likely due to poor target definition and also the variability of the target.

INTRODUCTION:

A matched filter is a spatial frequency filter designed to match an image with another in a set of very similar images. In addition, it can be used to locate objects within images. This makes it useful in character recognition, facial recognition, etc. It can be used to scan an image of text directly into a word processor for editing or used to recognize license plate numbers of cars in a parking lot or on the highway. It can also be used to calculate wind velocities from satellite imagery. A cloud formation can be selected in one image and used as the target in a matched filter system to locate the cloud formation in an image taken a few moments later. The wind velocity vector can be estimated using this technique.

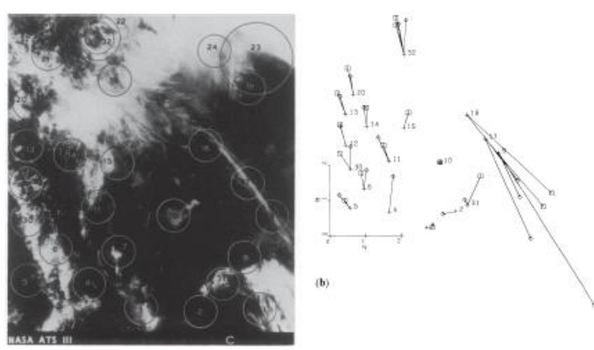


Figure 1: (left) satellite image of cloud formation; (right) wind velocities calculated from image.

The purpose of this project is to use the matched filter to locate alphanumeric characters in binary text inputs and to observe peaks in the output at the target locations. In addition, it will be tested using an actual photograph of the page of a book instead of simple binary text inputs. Finally the matched filter will be tested in facial recognition. An image from the "Where's Waldo" series will be used as the input with Waldo's face as the target.

The basic setup for the filter is a simple 4f-system in which the actual filter is placed in the Fourier plane, that is the back focal plane of the first lens and the front focal plane of the second. The input is placed at the front focal plane of the first lens and is illuminated with coherent monochromatic light.

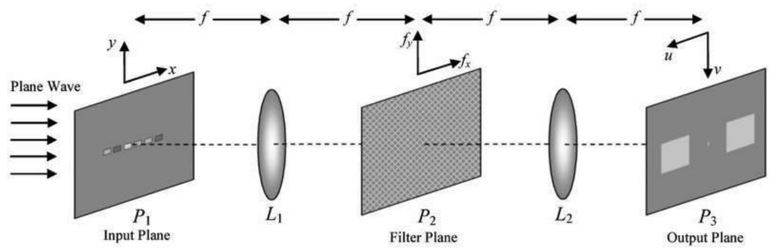


Figure 2: simple 4f system

THEORY:

How is a filter selected and defined for a specific target character? The resulting output plane of the 4f system with the filter in place should have a peak in intensity at the locations of the target character with a noisy, less intense background. Imagine a 4f system with a light meter placed in the output plane. Let the input be g(x,y). Passing through the first lens effectively takes the Fourier Transform of the input.

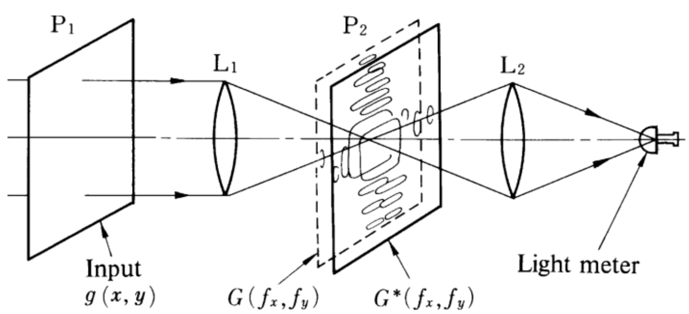


Figure 3: 4f system with matched filter

$$\mathcal{F}\{\text{input}\} = G(\xi,\psi) = |G(\xi,\psi)|\, e^{j\phi(\xi,\psi)}$$

The filter multiples this field distribution. Let the filter have an amplitude and phase transmission equal to $|G(\xi,\psi)|\, e^{-j\phi(\xi,\psi)}$ which is the complex conjugate $G^*(\xi,\psi)$. The resulting field is a plane wave with amplitude distribution $|G(\xi,\psi)|^2$. In passing through the second lens, this plane wave will converge directly into the light meter creating a peak in intensity. If any other filter function is used, the resulting wavefront will not be planar and the light will not converge neatly into the light meter giving a less intense reading. This setup can be used to match images from a set of similar images. If the Fourier transform of the target image is known, then, as each image is sent through the system, a peak in output will only be reached when the input matches with the filter.

Can this same filter design be used for character recognition? Imagine the 4f system again but without the light meter. The input is g(x,y) and is equal to the characters “a” and “b” on the y-axis separated by a distance 2l.

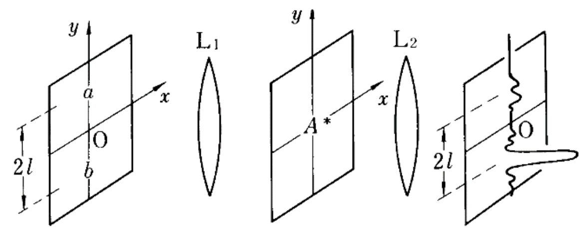


Figure 4: 4f system illustrating input and output relation of matched filter

$$g(x,y) = a(x,y-l) + b(x,y+l)$$

Let the target be the character “a” given by a(x,y) and $\mathcal{F}\{a(x,y)\} = A(\xi,\psi)$. The matched filter therefore be equal to $A^*(\xi,\psi)$ according to the preceding discussion. In the Fourier plane, the following product is acquired.

$$G(\xi,\psi)\, A^*(\xi,\psi)$$

The final lens takes the Fourier Transform again and we get the following convolution of Fourier Transforms in the output plane.

$$\mathcal{F}\{G(\xi,\psi)\}*\mathcal{F}\{A^*(\xi,\psi)\}$$

This can be simplified by using properties of the Fourier Transform as shown below.

$$\mathcal{F}\{\mathcal{F}\{g(x,y)\}\} = g(-x,-y) \tag{1}$$

$$\mathcal{F}\{g^*(x,y)\} = G^*(-\xi,-\psi) \tag{2}$$

Combining expressions (1) and (2), the convolution of the two Fourier Transforms in the output plane can be simplified to…

$$a^*(x,y)*a(-x,-y-l)+a^*(x,y)*b(-x,-y+l) = [a(x,y)*a^*(-x,-y-l)+a(x,y)*b^*(-x,-y+l)]^*.$$

Finally, there is a connection between convolution and correlation as shown below.

$$f(x,y)*g^*(-x,-y) = f(x,y)\bigstar g(x,y)$$

Using this relationship, the expression for the output can be simplified again to reach the final expression for the output of our matched filter 4f-system.

$$[a(x,y)\star a(x,y-l)+a(x,y)\star b(x,y+l)]^*$$

This is essentially the correlation of the input and target. These kinds of filters are often called correlation filters for this reason. This particular correlation will have a high peak at –l due to the auto-correlation of the two "a" transmission functions. There will also be smaller peak at +l due to the cross-correlation of the "a" and "b" transmission functions. Note that this output is inverted from the original orientation of the two letters. This is a result of the two subsequent Fourier Transforms.

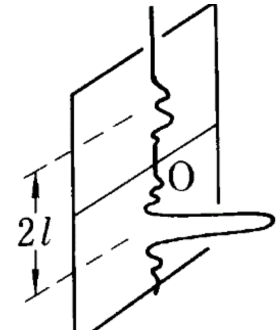


Figure 5: output of filter system. Note inversion.

The preceding filter design prescription will now be tested with variety of inputs. Each input will be saved as a 24-bit RGB bitmap image which will be converted into an 8-bit greyscale in the modeling software before passing it through the filter system.

<u>TEST #1:</u> Two Letter Binary Input

The first test will be on the same input used in the derivation in the Theory section, that is the letters "a" and "b" on the vertical axis. The output of which should be similar to Figure 5. Each step in the simulation will be illustrated for this test. Further tests will only show output.

The following input and target are imported to VirtualLab and converted to greyscales as shown in Figure 7. Simulating the presence of a lens, the forward transform is taken for both input and target. The target transform is then conjugated, and the two are multiplied to simulate the action of the filter.

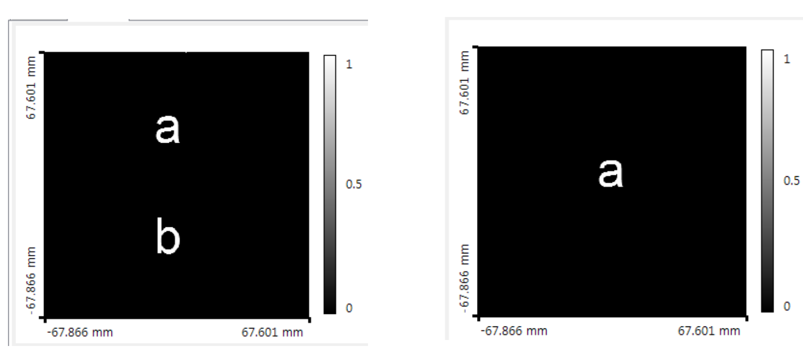


Figure 6: (left) input; (right) target.

Once this is done, another forward transform is taken simulating the second lens and the output is

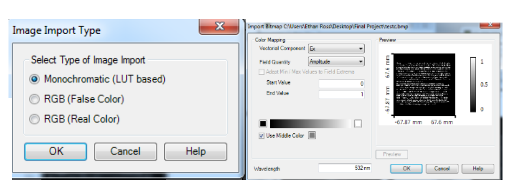
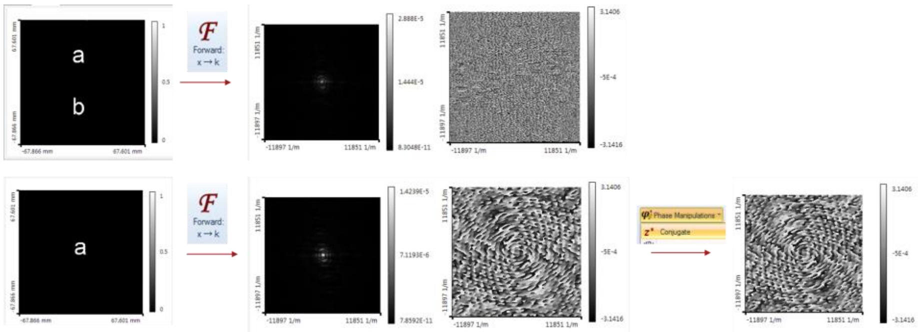


Figure 7: image import steps in VirtualLab

laterally displaced vertically and horizontally to account for the expected inversion. See Figure 8 for a graphical illustration of these steps in VirtualLab.



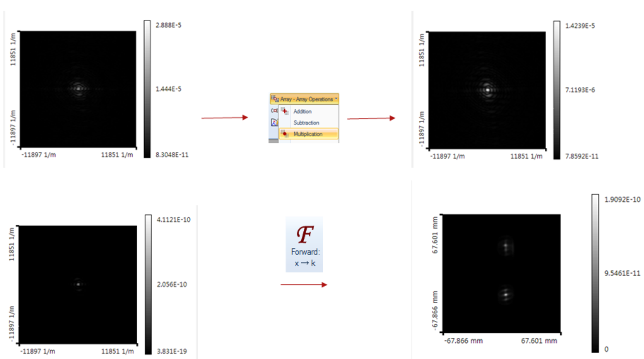


Figure 8: schematic of steps taken in VirtualLab; (line 1) FT of input; (line 2) FT of target and its conjugation; (line 3) multiply two FTs; (line 4) FT again

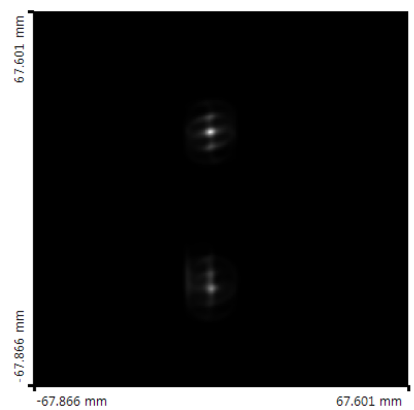
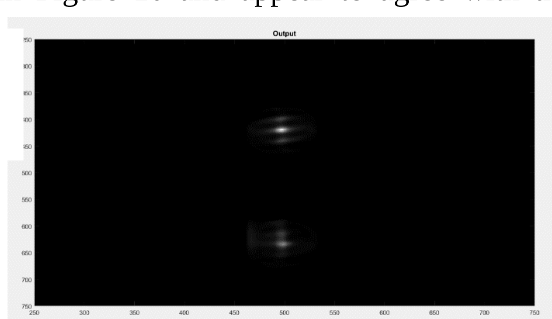


Figure 9: output in VL

The final output has two peaks as expected. One is more intense than the other and corresponds to the location of the target letter “a”.

We repeat the simulation in MatLAB for comparison. Instead of walking through each transform as in VirtualLab, the cross correlation between target and input was taken directly. (See Appendix for full code.) The results are shown in Figure 10 and appear to agree with the VirtualLAb results.

Figure 10: output in MatLAB



TEST #2: Two paragraph Binary Input, Single Character Recognition

Next a more complicated input will be tested. The input is a binary image of two paragraphs typed. The target is the capital letter “A” which appears seven times in the input. The output should therefore show seven sharp peaks in intensity over a noisy background. The output was simulated in VirtualLab and MatLAB.

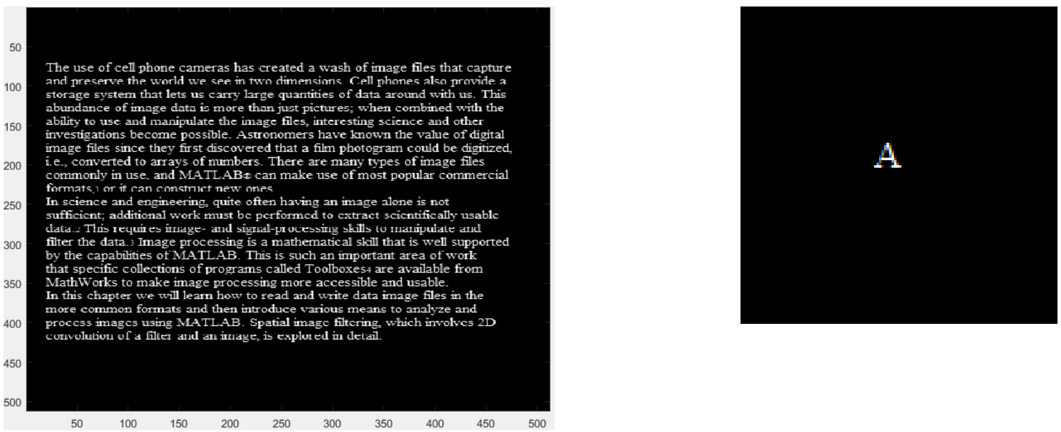


Figure 11: (left) input; (right) target

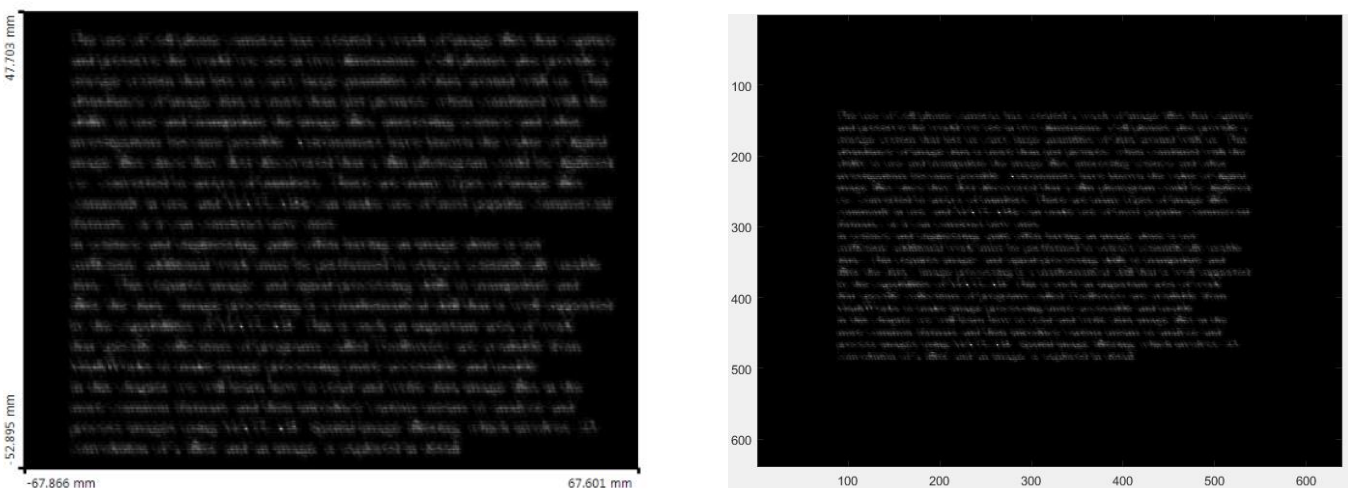


Figure 12: (left) output VL; (right) output MatLAB

Seven peaks are observed in both simulations at the expected locations.

TEST #3: Two Paragraph Input, Word Recognition

Next we test the same input but with a word as the target instead of a single character. The word "MatLAB" appears three times in the input. Three peaks are expected in the output. See Figure 14 for output.

The simulations in both programs generate three peaks over a noisy background, and these peaks correspond to the target locations.

Figure 13: target

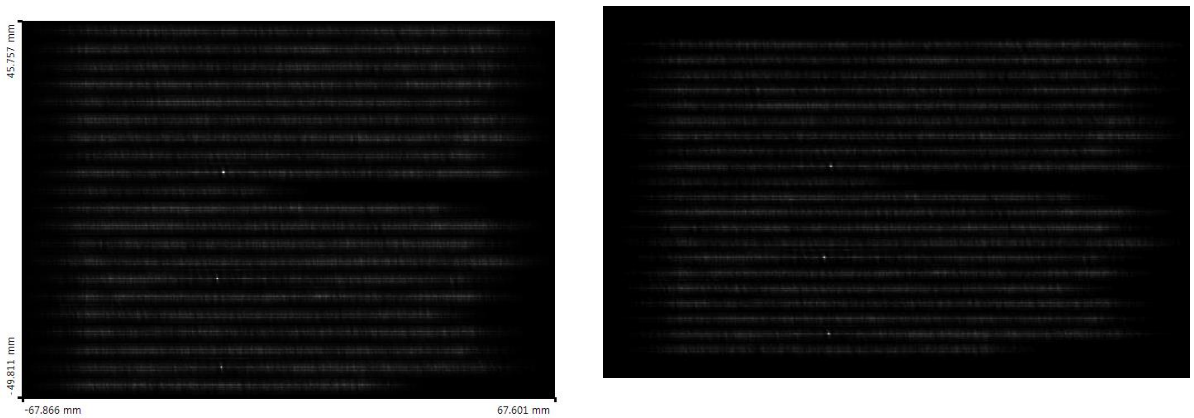


Figure 14: (left) output in VL; (right) output in MatLAB

TEST #4: Photographic Input, Single Character Recognition

Next the system is tested against a real photograph, not a binary transmission function. A photograph was taken of the page of a book and a segment of this was used as the input for this test. Target definition becomes complicated because of the noisy background of the input. In addition, complications arose in capturing the target with some background as both VirtualLab and MatLAB artificially darkened the background noise. See Figure 15.

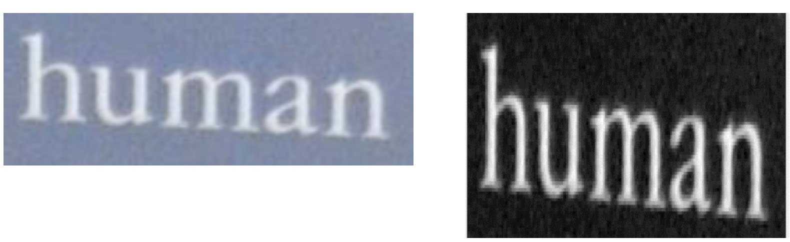
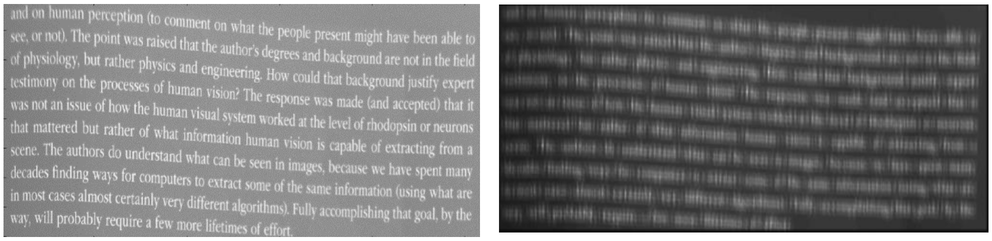


Figure 15: target complication when importing to VL and MatLAB. (left) original image; (right) after import.

In order to avoid this inexplicable complication, the target was traced and cut in Paint. This means no character can have a closed loop as this would inevitably include background noise. The target chosen was the letter "H". The input is shown in Figure 16 and the output in Figure 17.

Figure 16: target



and on human perception (to comment on what the people present might have been able to see, or not). The point was raised that the author's degrees and background are not in the field of physiology, but rather physics and engineering. How could that background justify expert testimony on the processes of human vision? The response was made (and accepted) that it was not an issue of how the human visual system worked at the level of rhodopsin or neurons that mattered but rather of what information human vision is capable of extracting from a scene. The authors do understand what can be seen in images, because we have spent many decades finding ways for computers to extract some of the same information (using what are in most cases almost certainly very different algorithms). Fully accomplishing that goal, by the way, will probably require a few more lifetimes of effort.

Figure 17: (left) input; (right) output in MatLAB.

A peak is observed at the location of the letter "H" in the third line but its intensity is not as distinct as previous binary input results. This is expected with the more noisy input but likely could be improved with better target definition. There also appear identically bright peaks throughout the image, for example, at the locations of the capital "T". This also could be improved with better target definition.

<u>TEST #5:</u> Facial Recognition

Finally, the filter's mettle is tested against a facial recognition problem. *Where's Waldo* is a popular series in which readers attempt to locate Waldo in a large and densely populated image. Waldo is usually small and very hard to discern. The filter could potentially produce a bright spot at his location, thereby negating the need to find him by eye and the associated mental torture in doing so.

However, facial recognition introduces new problems and makes old problems much more difficult. As before and unlike the binary inputs, background is complications in defining the target, but that complication is now compounded by the variability of Waldo, his expression, the orientation of his head, and his overall size. Also, each Waldo appears in a different environment meaning his pixel content is much different. Waldo on a beach will not be the same as Waldo indoors. This means that by the method used in this project a myriad of Waldo filters must be created to account for each of these variations. This will be a high number.

This test quickly demonstrates that a peak is generated at Waldo's location. Beyond this, nothing further testing was done. See Figures on following page for results.

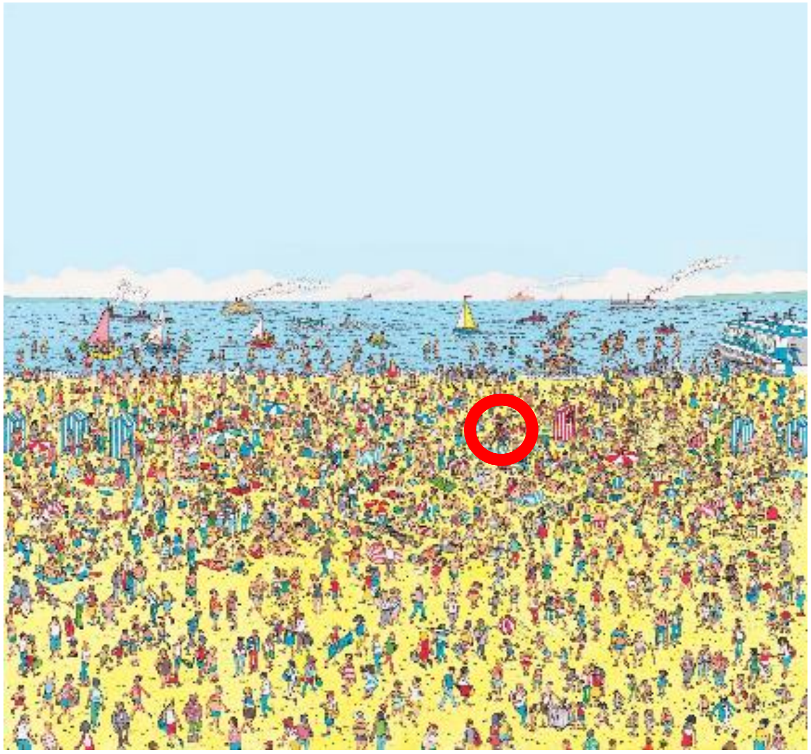


Figure 18: red circle is Waldo's location

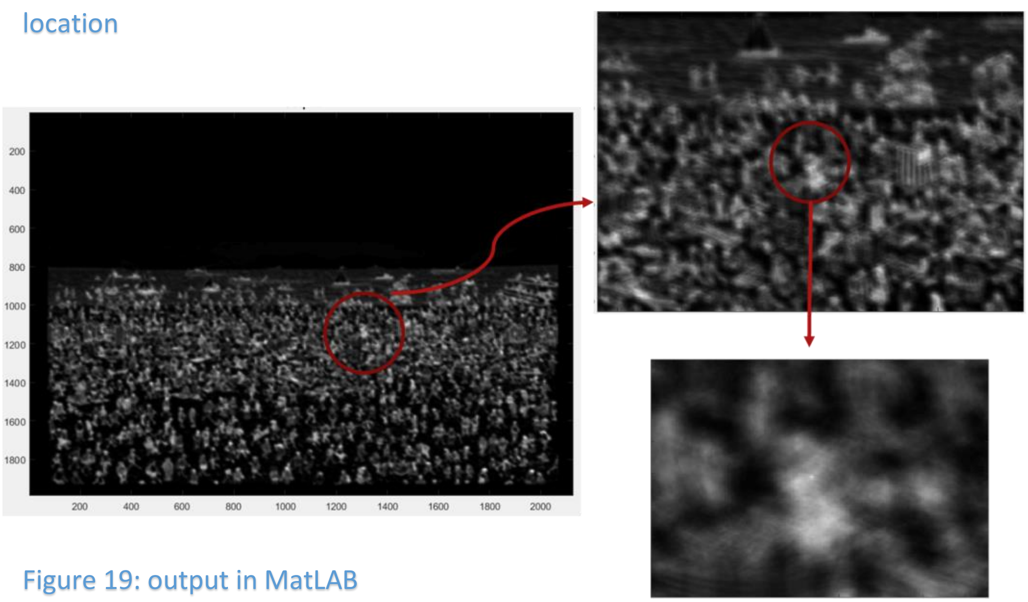


Figure 19: output in MatLAB

CONCLUSION:

This project has demonstrated a matched filter 4f-system that is efficient in locating alphanumeric characters in a binary input. More noisy text input also generates peaks but less distinctly due to background noise and target definition complications. The matched filter also has potential in facial recognition problems. However, this method requires the use of many filters for a single face. Further research should be in the area of target definition and image preparation.

RESOURCE:

Iizuka, Keigo. *Engineering Optics*. New York, NY: Springer, 2010. Print.

APPENDEX: MatLAB Code

```
% Skip Fourier transform (4f system) method.  Just take correlation of
% object with target.  The result should be the same
clear;clc;

%Read in target
target=imread('target_bookpage_human.bmp','bmp');
target=rgb2gray(target);
target=double(target)/255;

%Read in image
image=imread('bookpage1.bmp','bmp');
image=rgb2gray(image);
image=double(image)/255;

%Take cross-correlation
output=xcorr2(image,target);
[num,idx]=max(output(:)); %finds indices of output maximum
[x,y]=ind2sub(size(output),idx);

figure(1)
imagesc(target)
colormap('gray'); title('Target');

figure(2)
imagesc(image)
colormap('gray'); title('Object');

figure(3)
imagesc(abs(output).^4); title('Output');
colormap gray
```